メカトロニクス応用実験

1. 実験目的

メカトロニクスは機械工学と電子・情報工学との融合技術である．コンピュータを利用して機械を制御しようとするとき，機械からコンピュータにデータを取り入れ，またコンピュータから機械に処理結果を出力しなくてはならない．このときコンピュータと機械のそれぞれ異なった現象を関係付けるものがインタフェイスである．インタフェイスとはコンピュータと機械の間ばかりでなく，人とコンピュータ，人と機械の間の情報や意思の疎通をも含めた広い意味で用いられる用語である．**本講においては，メカトロニクスを構成するコンピュータとセンサやアクチュエータを結びつけるインタフェイス技術と制御技術を，実機を通して学ぶことを目的とする．**

2. インタフェイス回路と制御機器

2．1 コンピュータ制御の基本

コンピュータ制御の構成図を Fig. 1 に示す．本実験では，パソコンの USB ポートに接続された I/O ボード（USB コントローラ）を通して，LED 表示ボードやステッピングモータなどの機器をコンピュータ制御する．I/O ボードはパソコンで制御される機器と CPU との間のデータの中継の役割を受け持つため，入出力インタフェイスと呼ばれている．

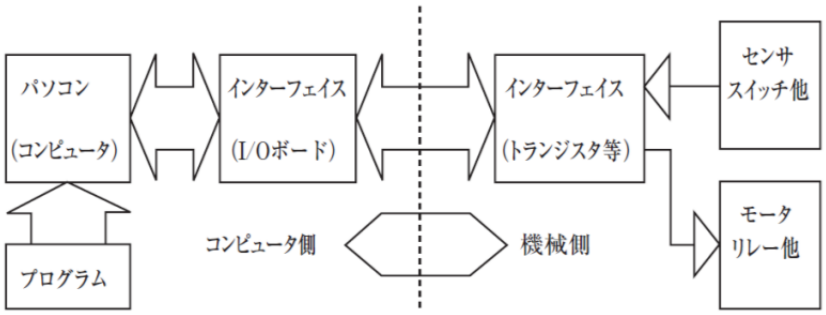


図1 パソコン計測と制御の構成図

2．2 使用する実験器具 USB-EXE

Fig.2 には実験装置の外観を示す．コンピュータのプログラムで何をするか，このプログラムの動作対象が制御機器である．一般にモータ，センサ，スイッチ，表示装置等で構成されるが，モータやセンサは直接 I/O ボードと接続できない．このため，インタフェイス回路を介して I/O ボードと接続する．インタフェイス回路にはトランジスタや IC が使用されており，USB-EXE ではモータ類とインタフェイス回路が一体となって組み込まれている．プログラム作成時には制御する機器の構成や回路の働きを理解しておく必要がある．また，コンピュータとどのように接続されているかについても理解しておく必要がある．例えば，I/O ボードの何番のポートに接続されているのか，また，何番目のビットと接続されているのかなどである．制御機器によって 1 ビットの場合や 1 つのポート全ビットを使用する場合もある．USBI/O ボードは 8 ビットのポートが 4 つあり，それぞれ PIO0, PIO1, PIO2, PIO3 と呼ぶ．

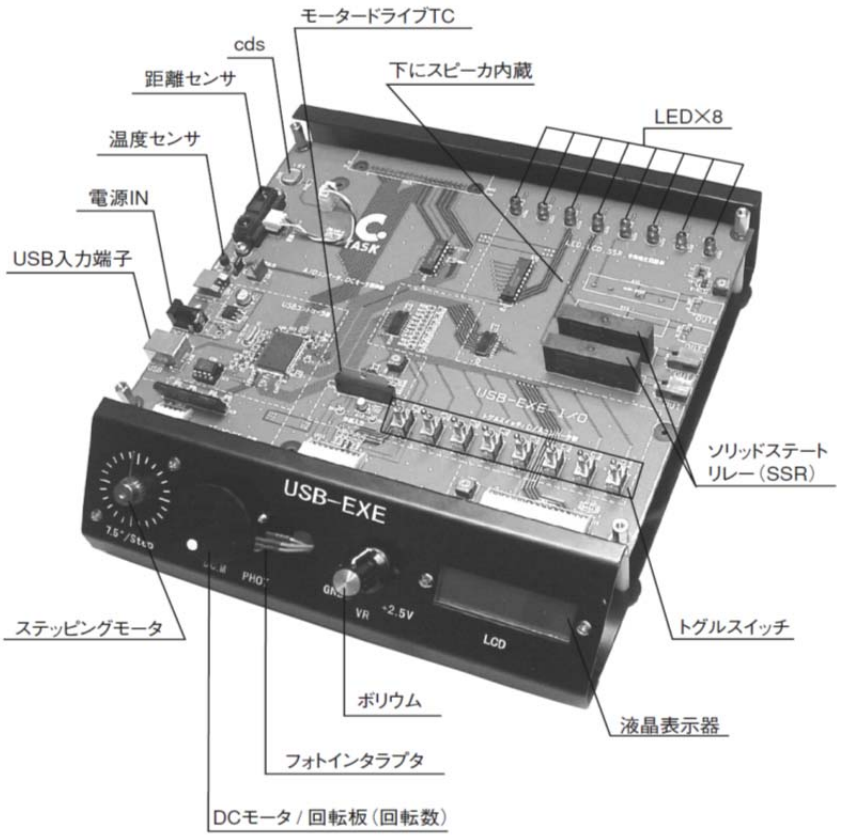


図2 メカトロニクス実験装置 USB-EXE

2. 2. 1 USB-EXE のPIO0

USB I/O の出力は 3.3V である．そこで電圧と電流を確保するため 74HC540 のインバータ IC を通して LED と接続している．パソコンより"1"を出力すると対応する LED が点灯する．

2. 2. 2 USB-EXE のPIO1

ラダー型の D/A コンバータとトグルスイッチが接続してある．ポート PIO1 に 0 から 255 を出力すると D/A コンバータの出力電圧は 0V から約 3.3V となる．D/A コンバータの出力は直流モータのドライブ IC である TA7291P に加えている．D/A コンバータの出力で直流モータの回転速度制御ができる．直流モータの回転方向の制御はポート PIO2 からの出力で行っている．

2. 2. 3 USB-EXE のPIO2

PIO2 にはシリアル型の A/D コンバータを接続している．A/D コンバータはアナログ信号をディジタル信号に変換する回路である．A/D コンバータには 4 回路の入力があり，どの入力を使用するかはプログラムで切り替える．シリアル型の A/D コンバータは I/O ボードとの接続に必要なビット数は少なくてよいが，プログラムがやや複雑になる．A/D コンバータの入力に可変抵抗，CdS（明るさで抵抗値が変化するセンサ），距離センサモジュール（センサとの距離で出力電圧が変化する），温度センサが取り付けてある．ここで使用している A/D コンバータは 0V から 2.5V のアナログ入力が 0 から 255 のディジタル値に変換される．

フォトインタラプタは発光ダイオードとフォトトランジスタを組み合わせたもので，この間に光を遮断するものを入れる．実験システムでは直流モータに取り付けた円盤に穴を開け，この穴が来たときに光が通るようにしている．

3. 実験　第一週目　AD変換

第 1 週は AD 変換，第 2 週は DC モータの制御の実験を行い，センサやアクチュエータを制御するソフトウェアを通してメカトロニクスの基本動作を実践しながら理解する．まず，付録に示した C#のプログラムを実行して基本動作を確認する．その後，目的に応じてプログラムの書き換えを行い，それぞれの動作を確認しながらソフトウェアを作成していく．

3. 1　AD変換･DA変換

AD 変換とは，アナログ信号をディジタル信号に変換することである．電気信号においてアナログは連続的に変化する時間であり，ディジタルは離散的な情報である．アナログ信号は直感的に分かりやすいがノイズによる影響を受けやすく劣化しやすい．一方で、ディジタル信号はノイズに強く、信号を計算機で扱えるようになる．センサで得られたアナログ信号を計算機で扱う時は，必ずディジタル信号に変換する必要がある．AD 変換では標本化と量子化両方の処理が必要である．連続信号を一定の間隔をおいて測定することを標本化（サンプリング），測定値を離散的な数値で表すことを量子化と言う．AD 変換にはパラレル方式とシリアル方式がある．パラレル方式のほうが取り扱いも簡単で変換速度も速いが，コンピュータとの接続に多くのビット数が必要．シリアル方式では 1 ビットずつ 8 回読みとったところで一つのデータとなる．

図 3　アナログ温度計(FICST 社)

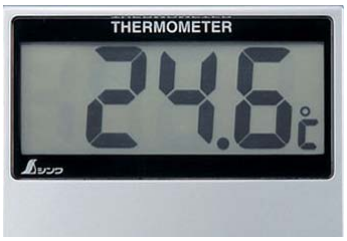


図 4　デジタル温度計（シンワ社）

3. 2　標本化誤差と量子化誤差

まず、標本化では、一定の間隔でアナログ量が間引きされる。この間隔をサンプリング間隔（サンプリング時間 Δt、サンプリング間隔 Δx)と呼ぶ。間引きされたアナログ値はその近辺（時間的 or 空間的）の代表値であるため、その間にあった値が抜け落ちてしまう。これを標本化誤差と呼ぶ。サンプリング間隔（時間）を狭めればこの誤差は小さくなる。

量子化誤差とは，アナログ値をディジタル値に変換する際に生じる数値的な丸め誤差である（図４　右 参照）．例えば，測定範囲が-50℃～206℃のアナログの温度計があるとする。ある時刻における温度を 8 ビットのディジタル表示に変換する場合，-50℃～206℃の範囲を 2^8=256 分割して表記することになる．目盛の間隔は 1.0℃となるが，最悪の場合(目盛の中間にアナログ値が位置している場合)は±0.5℃の誤差が発生することになる．

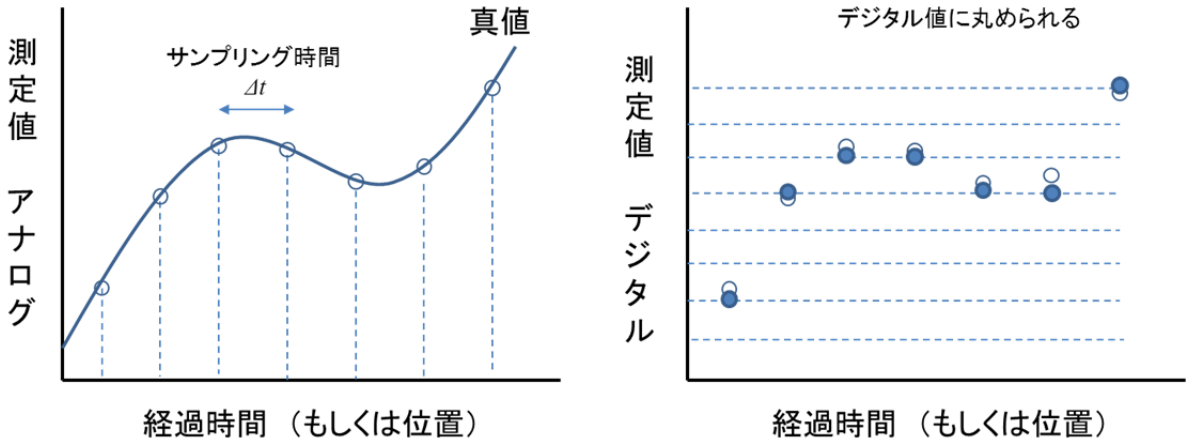


図 4(a) 標本化と量子化 （最終的に右図の●がディジタル化された情報である）

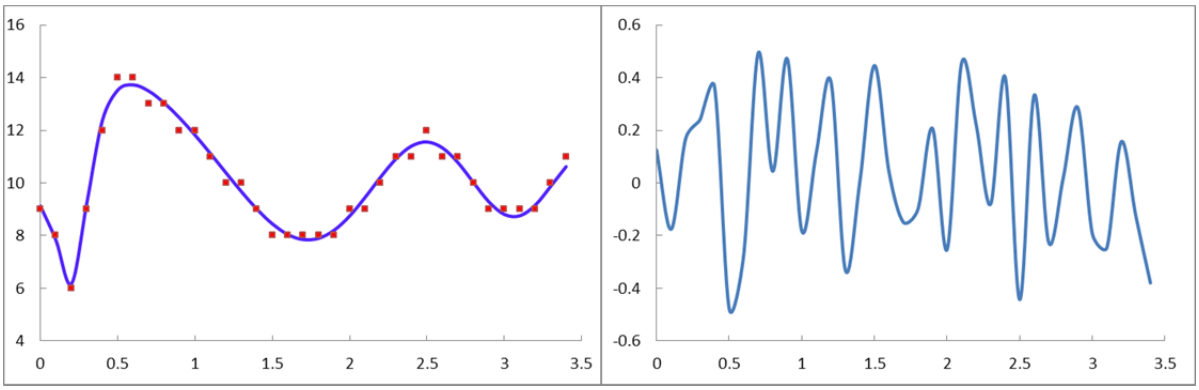


図 4(b) 量子化誤差

3. 3 AD変換器の回路図

逐次比較 AD コンバータでは，図 5 の右図にあるアナログスイッチ 1 を閉じると電圧 Es は $V_{ref}/2$ になる．もし $Es > V_{in}$ ならこのスイッチ 1 は閉じたままとし，そうでない場合はスイッチを開く．次にアナログスイッチ 2 を閉じると Es は$(V_{ref}/2 + V_{ref}/4)$になる．ここでも Es と V_{in} を比較し，上記と同様の操作を行う．8bit の AD 変換の場合，図 5 のようなラダー回路でスイッチを 8 個付けたものを作り，操作を 8 回繰り返していくことになる．8 回繰り返した後，スイッチ n について，PA_nを閉じているなら 1，開いているなら 0 とする．これを順に横に並べたものが求めたいディジタル値となる．

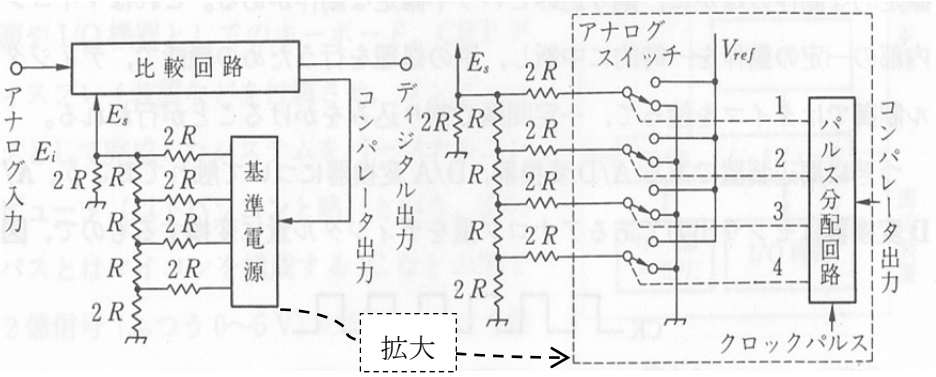


図 5 逐次比較 AD 変換機の原理（4bit の場合）

変換手順例

アナログフルスケール電圧を V_{ref}=5.00[V]，入力電圧を V_{in}=3.54[V]としたときの変換過程

①スイッチ 1 を閉じると，Es=V_{ref}/2=2.5[V]となる．
　V_{in} > Es なので，スイッチ 1 は今後常に閉じておく，PA_1=1
②この状態でスイッチ 2 を閉じると Es=V_{ref}/2+ V_{ref}/4=2.5 + 1.25=3.75[V]となる．
　V_{in} < Es なので，スイッチ 2 は今後常に開いておく，PA_2=0
③同様にスイッチ 3 を閉じると Es=V_{ref}/2+ V_{ref}/8=2.5 + 0.625=3.125[V]となる．
　V_{in} > Es なので，スイッチ 3 は今後常に閉じておく，PA_3=1
④スイッチ 4 を閉じると Es=V_{ref}/2+ V_{ref}/8+V_{ref}/16=3.4375[V]となる．
　V_{in} > Es なので，スイッチ 4 は今後常に閉じておく，PA_4=1
⑤スイッチ 5 を閉じると Es=V_{ref}/2+ V_{ref}/8+V_{ref}/16+ V_{ref} /32=3.59375[V]となる．
　V_{in} < Es なので，スイッチ 5 は今後常に開いておく，PA_5=0
⑥スイッチ 6 を閉じると Es= V_{ref}/2+ V_{ref}/8+V_{ref}/16+ V_{ref} /64=3.515625[V]となる．
　V_{in} > Es なので，スイッチ 6 は今後常に閉じておく，PA_6=1
⑦スイッチ 7 を閉じると，Es= V_{ref}/2+ V_{ref}/8+V_{ref}/16+ V_{ref} /64+ V_{ref} /128=3.5546875[V]となる．
　V_{in} > Es なので，スイッチ 7 は今後常に開いておく，PA_7=0
⑧スイッチ 8 を閉じると Es= V_{ref}/2+ V_{ref}/8+V_{ref}/16+ V_{ref} /64+ V_{ref} /256=3.53515625[V]となる．
　V_{in} < Es なので，　　PA_8=1
変換結果，ディジタル出力は“1011 0101”となる．
一方，DA 変換の場合も同様にラダー回路を使用する．DA 変換では 8 ビットのディジタル信号の入力により 8 つのスイッチの開閉が一意に決まる．これにより回路出力電圧を 256 段階で操作できることになる．一般式は次式で与えられる．

$$V_{\mathrm{out}} \cong V_{\mathrm{ref}}\left(\frac{PA_1}{2}+\frac{PA_2}{4}+\frac{PA_3}{8}+\frac{PA_4}{16}+\frac{PA_5}{32}+\frac{PA_6}{64}+\frac{PA_7}{128}+\frac{PA_8}{256}\right)$$

(ここで PA0～PA7 は 0 か 1 (Low or High)である．)
仮にフルスケール電圧V_{ref} =5[V]において，3.54[V]の電圧をモータに与えたい時，8bit のディジタル信号 10110101 を DA 変換器に送信すればよいことになる．
上の式では 5×(1/2+1/8+1/16+1/64+1/256) = 3.535[V] ≅ 3.54[V]となる．

3．4 AD変換とPCとの通信

本実験で使用する USBEXE の AD 変換器はシリアル型であり，4 つのチャンネルを有する．チャンネル 1 つにセンサが 1 つ対応づけられる．同時に複数のセンサ情報を変換することはできないことに注意．従って，AD 変換器から特定のセンサのディジタル量を受信するプログラミングをする際に，そのセンサが繋がっているチャンネルを指定しなければならない．

チャンネル 0　温度センサ　（今回は使用しないので無視してよい）
チャンネル 1　距離センサ
チャンネル 2　光センサ　（CdS センサ）(Cd：カドミウム, S：硫黄)
チャンネル 3　ボリウム（可変抵抗）

4.　使用するセンサー

4.１　ボリウム（可変抵抗）

本実験で使用する USB-EXE では，可変抵抗はボリュームつまみに応じ抵抗値(0～10kΩ)を変化させ出力電圧を変化させることができる．この回路図を図６に示す．Vcc=5.0[V]が供給されており，可変抵抗のボリュームを操作することにより出力電圧は 0.0V から 2.5V まで変化する．この出力電圧 Vout は AD 変換器に送られ 256 段階でディジタル値に変換される．

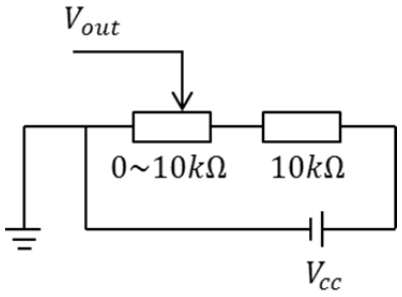


図６　可変抵抗器の回路図

4.２　距離センサ

今回の実験では、PSD（Position Sensitive Detector）を使用している SHARP 社製の GP2Y0A21YK を用いる．PSD 素子は抵抗体とフォト・ダイオードを組み合わせた素子で，フォト・ダイオードを横に長く延ばした構造になっている．光が入射するとフォト・ダイオードと同様に電流が流れる．光の入射位置から出力端子までの距離に比例して抵抗が変化するため，出力端子を 2 つ設けることで，それぞれの出力端子の電位差から三角測量の原理により距離を測定できる．このため，出力電圧と対象物との距離はほぼ反比例する．

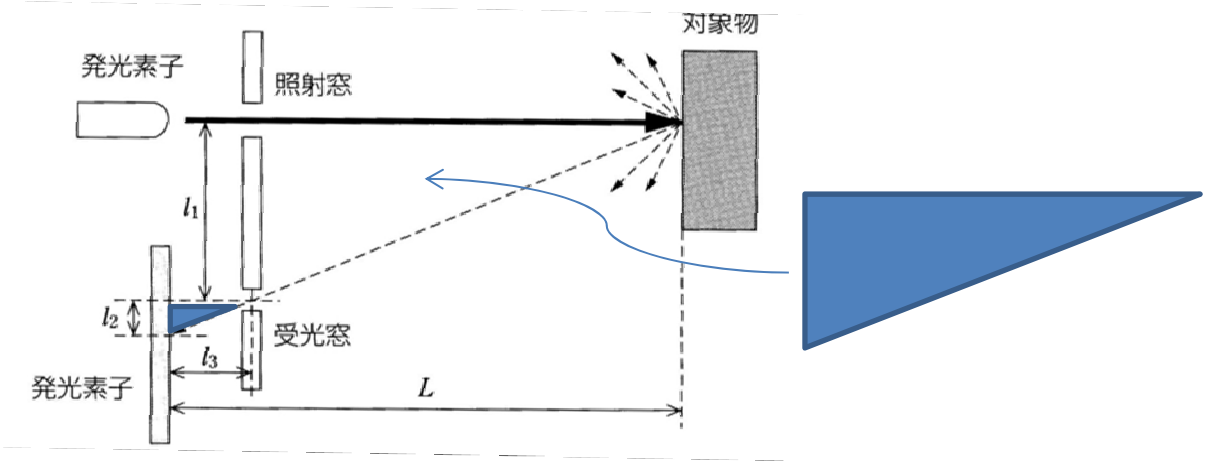


図７　距離センサの測定原理（三角測量）

$$L = \frac{l_1 l_3}{l_2}$$

($l_2 \propto$ 出力電圧，l_1 と l_3 は定数)

→ 対象物までの距離 L とセンサ出力電圧は反比例の関係にある

3. 2. 3　光センサ(CdS センサ)

本実験システムでは光センサとして CdS センサを使用している．CdS とは硫化カドミウムのことで当たる光の量にしたがって抵抗値が変化する．つまり CdS センサとはそれをモジュール化して，光が入射することにより，電子が伝導体に励起され，電気抵抗値が低下する性質を利用したセンサである．

3. 3　プログラミング

今回の実験では Microsoft 社の Visual C#を使用する．この言語は従来の C 言語に近く，C 言語の経験者に適する．制御プログラムの第一歩はデータの入力と出力である．制御機器の仕様と何をどう制御するかわかれば，プログラムを組むのは簡単である．

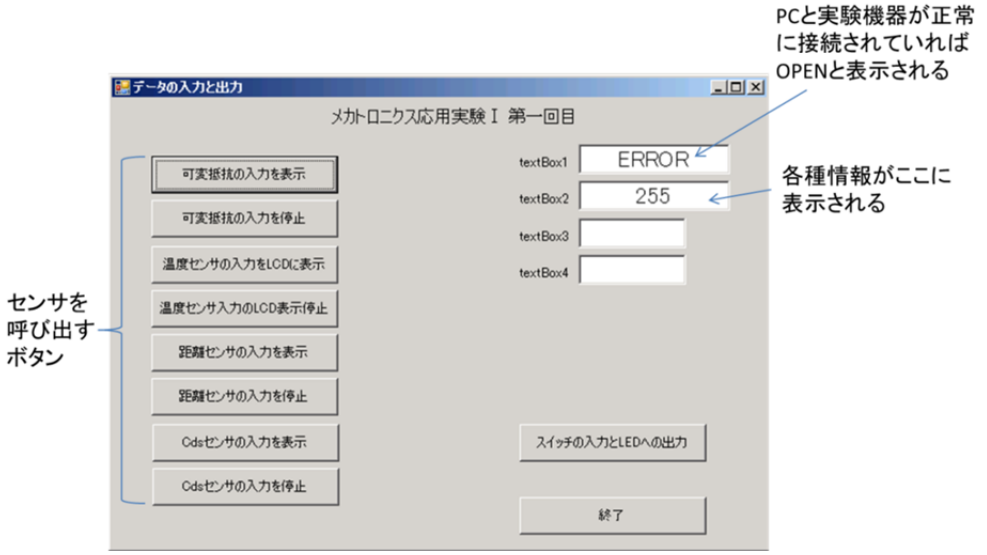
4. 実験（1週目）

＊動作テスト

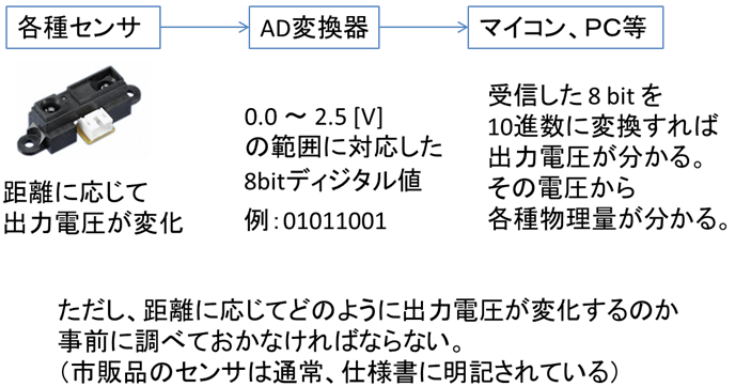
デスクトップ上のフォルダ　”メカトロ応用一回目”　を開き YN_1.sln をダブルクリックして Visual Studio C#のプログラムを立ち上げる．画面右のソリューションエクスプローラーの枠にある From1.cs を右クリック，コードの表示をクリックする．もう一度 From1.cs を右クリック，デザイナの表示をクリックする．プログラムをデバッグ無しで実行する．(F5 キーで可能)

ダイアログボックスが表示される。右上のボックス内に「OPEN」と表示されれば USB-EXE と PC が正常に接続していることを意味し、「ERROR」と出れば接続不良であることを意味する。「ERROR」と出た場合は、右下の「終了」ボタンで一旦画面を閉じ、USB-EXE の電源コードと USB を外し、再度、接続しプログラムを実行(F5 キー)する。

「OPEN」になったら、左上の「可変抵抗の入力を表示」ボタンを押してみる．
すると、右上の下のボックスに数値が表示される。
ボリウムつまみを回すとこの数値が変化することを確認する。



図：プログラムを実行した後に表示される画面



図：センサを用いて物理量を取得する流れ

第 1 週目：実験課題

課題 1． ボリウムは、左に回しきった時 0 [kΩ]、右に回しきった時 10 [kΩ]であり，その間の抵抗値は線形に変化すると仮定する。また、抵抗値に比例して電圧(0.0V～2.5V)が出力され、AD 変換器に送られている。そして、AD 変換器の 8 bit のディジタル信号が 10 進数(0～255)に変換され、ボックス内に表示されるようになっている．

そこで，「可変抵抗の入力を表示」ボタンを押した時、ボックス内に可変抵抗値が［kΩ］単位で表示されるよう，プログラムを書き換えなさい．

課題 2． プログラム内の///可変抵抗の入力///の 3 行を参考に、「距離センサの入力を開始」ボタンを押した時に距離に応じてデジタル値がボックス内に表示されるようにプログラムを編集しなさい。（チャンネルの設定に注意すること）

実験器に搭載されている距離センサの測定範囲は[0.1 m ～0.8 m] である。この範囲内で、距離センサの上方に 0.1 m 間隔で平らな物体をかざし、その時のデジタル値を下の表に記録しなさい．

表：計測結果

(a) 実際の距離 [m]	(b) デジタル値	(a)×(b)
0.1		
0.2		
0.3		
0.4		
0.5		
0.6		
0.7		
0.8		

↑の平均値を計算→＿＿＿＿＿＿

6 ページ 4.2 章より、実際の距離と出力電圧は反比例の関係にあるので、デジタル値と実際の距離も反比例の関係にある。

出力電圧のデジタル値＝**定数**÷実際の距離

この**定数**を上で求めた平均値で近似し、課題 1 と同様に、距離の値が[m]単位でボックスに表示されるよう、プログラムを編集しなさい。

課題 3． 光センサについて、簡易照度計を用いて同様の作業を行いなさい。単位は lx を用いること

課題 4． 以下の命令文は C#言語においてテキストファイルに指定した変数内の数値を書きだすプログラムである．これをプログラム内の適切な位置に挿入すると，各々のセンサの出力が時系列でテキストファイルに記録される。

```
////////////////////////////////////////////////////////////////////////////////
StreamWriter Writer =
new StreamWriter(@"C:¥Test.txt", true);
Writer.WriteLine("{0}", v);
Writer.Close();
////////////////////////////////////////////////////////////////////////////////
```

どのセンサの記録か、分かりやすいようにファイル名を変える。

記録される数値。ここもセンサに合わせて変える。

記録の際に失敗し記録し直す場合、名前を変えずに次の記録を取ると，前のデータの終了した所から次のデータがつぎ足される形で書き込まれるので注意．前のファイルを破棄するか、ファイル名を新しくかえる。C:¥kyori.txt 失敗→ C:¥kyori2.txt 失敗→ C:¥kyori3.txt 等と。

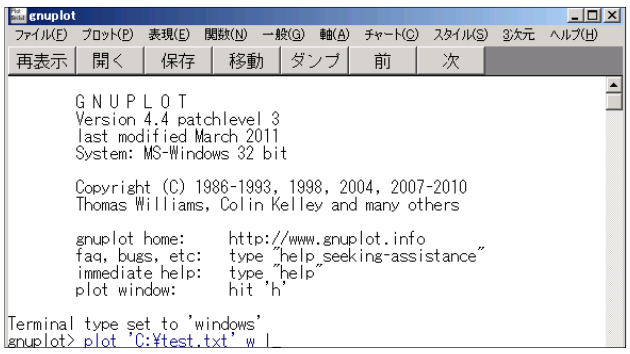
1 つのセンサにつき、1 つの計測記録（ファイル）を作成する。計測結果の確認は gnuplot というフリーソフトを用いて確認する。

・デスクトップ上にある wgnulot.exe を立ち上げる．

・plot□'C:¥test.txt'□w□l（□はスペース、最後の文字はエル）と打ち enter を押すと C:¥にある test.txt をグラフ表示する．(グラフは保存する必要はない、確認のための作業である)

・マウスは効かないので、カーソルキーだけで編集すること

・カーソルキーの上下を押すと、コマンド履歴が出てくる。うまく利用すれば plot□'C:¥test.txt'□w□l の test の部分だけを編集し、enter を押すだけでグラフが表示される。



第一週目は、3 種類のセンサ情報を記録し、グラフで確認できたら終了とする。

4.2 第2週目

2週目はDCモータの制御実験，及び制御に必要なプログラムの構造を理解する．DCモータの回転速度をリアルタイムで計測し，それを元にモータに与える電圧をコントロールすることで，目標の回転速度で回り続けるようフィードバック制御するプログラムを実装する．モータに印加する電圧，制御する回転速度はアナログ量であるが，コントローラの制御演算(C#のプログラム)ではディジタルであるためディジタル制御を行うことになる．しかし，速度の計測においてはそのアナログ量は実際の AD 変換器には通さず，サンプリングにより代用することになる．モータに与える印加電圧はモータドライバの特性上 DA 変換器を通して DA 変換を行わなければならない．

4.2.1 DCモータ

DC モータとは直流(Direct Current)モータのことであり，起動トルクが大きい，印加電圧に対して直線的に回転速度が比例する，入力電流に対し出力トルクが直線的に比例する，低価格であるという特徴がある．USB-EXE においては，I/O ポートから 10 進数で 0 から 255 を DA 変換器に送信し，0V から 3.32V の電圧に変換された電圧がモータに与えられる．つまり，PC 側ではモータの速度を 0~255 のディジタル値で制御することになる．

＊DC モータの速度制御を行う手法として入力電圧を変える方法(PAM)と通電時間を変化させる方法（PWM）があるが，本実験では前者の手法でモータの速度制御を行う．

＊PAM(Pulse Amplitude Modulation）パルス振幅変調

＊PWM(Pulse Width Modulation） ）パルス幅変調

PWM とは制御対象（モータ）への通電を短い周期で ON/OFF させ，一周期の中で ON/OFF の時間が占める時間の比（デューティー比）を変化させてモータへのエネルギーを制御する方法である．

4.2.3 フォトインタラプタによるDCモータの回転速度の計測

直流モータの軸には穴の開いた円盤が取り付けられている．この円盤が回転数を透過型フォトインタラプタと呼ばれるセンサで読み取りモータの回転速度を計測する．発光側に赤外発光ダイオード，受光側にシリコンフォトトランジスタを使用している汎用的な透過型フォトインタラプタである．発光側からでた光が円盤の穴が通過するたびに受光側に光が届く．Windows の OS の基では通常割り込みは使用出来ないため，受光側のセンサが感じたら次に感じなくなるまで待つ(穴を通過し終わるまで)．プログラム上ではこの時点で回転数を保存する変数を準備しておき，回転数をカウントアップしていく．

このプログラムが一定回数ループした所で速度の変数データの値を回転速度としている．制御プログラムを実行している間は CPU 速度一定と仮定すると，サンプリング時間⊿t も一定とみなせる．時刻 t[s]から⊿t[s]におけるモータの回転数 N を検出，回転速度 v[rps]は v=N/⊿t により求められる．本実験では⊿t＝1.0[s]としている．

プログラムでは pio2_in(0) & 0x20 が LOW のとき，円盤の穴を検知している．

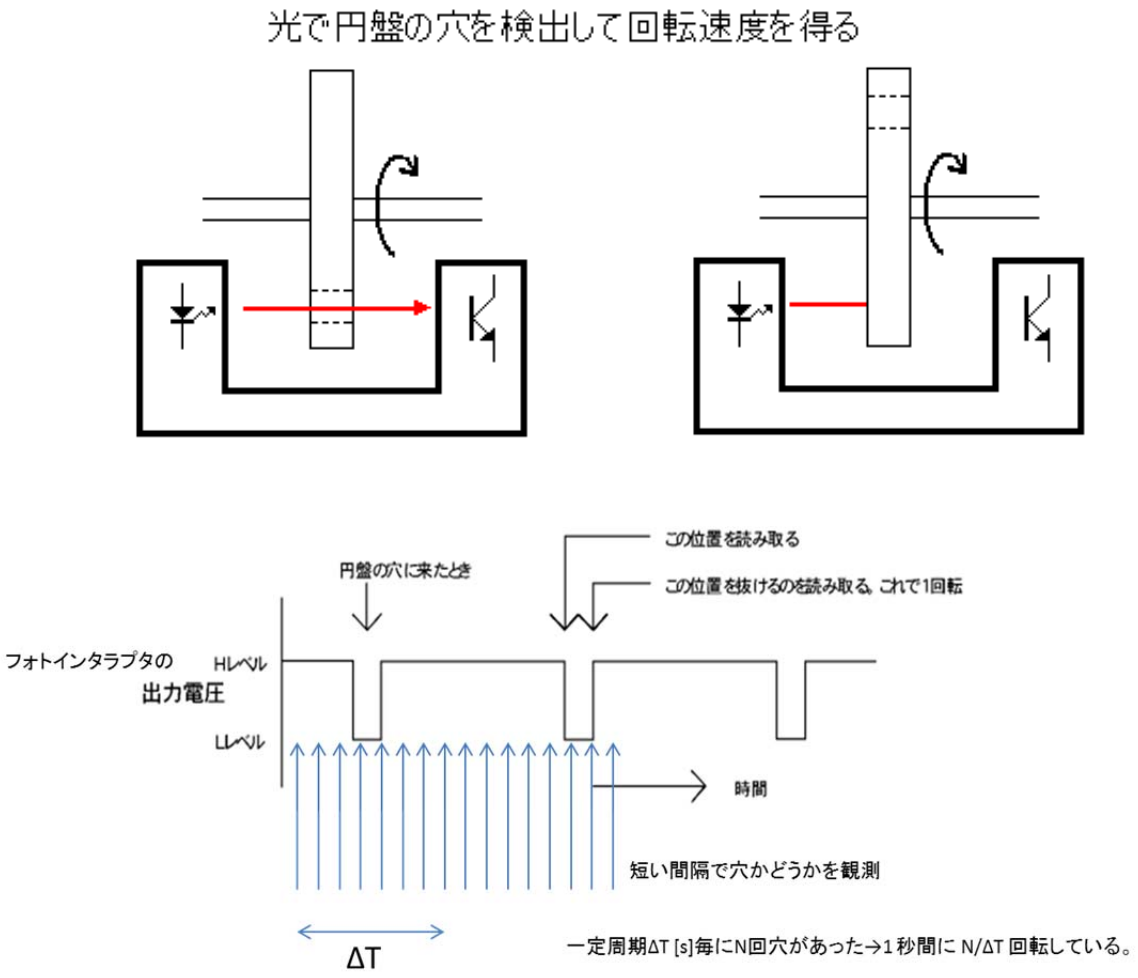


図9　フォトインタラプタによる回転速度の検出

4. 3 DCモータの速度制御

DC モータの速度制御とは，モータの回転速度を目標値(目標回転速度)に一致させることをいう．これは工作機械の送り速度の制御等に用いられている．今回の実験では、回転速度と目標値の差をフィードバックし， DC モータの速度を PI 制御により速度を制御する．

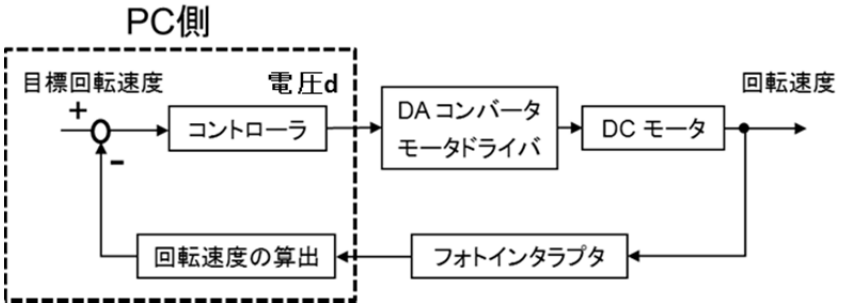


図10　DC モータの制御システム

4. 3. 1 DCモータの運動方程式

モータの回路方程式は以下のように表される。

$$E(t) = \mathrm{R}i(t) + \mathrm{L}\frac{di(t)}{dt} + \mathrm{K_v}\omega(t) \quad (1)$$

一方、モータの運動方程式は以下のようにあらわされる。（粘性抵抗、負荷トルクは無視）

$$J\frac{d\omega(t)}{dt} = \tau = \mathrm{K_A}i(t) \qquad (2)$$

E[V]は供給電圧、R[Ω]はモータの内部抵抗、i [A]はモータに流れる電流、L[H]はモータのインダクタンス、Kv[Vs/rad]はモータの逆起電力定数、ω[rad/s]は円盤の回転速度、J はモータ軸と回転円盤の慣性モーメントを表している。

両式をラプラス変換して i を消去し、入力を E、出力を ω とすると次式が得られる。

$$\omega(s) = G(s)E(s) = \frac{\mathrm{K_A}}{(\mathrm{JL}s^2 + \mathrm{JR}s + \mathrm{K_A K_v})}E(s) \qquad (3)$$

4. 3. 1 PI制御とディジタル制御

モータの運動モデルは既に示したように、電圧の入力に対し回転速度を出力する。すなわち速度を制御するには電圧を制御すればいい。PI 制御においては図 11 のコントローラ部を実装することにより、モータの目標回転速度と出力されている回転速度の差から電圧を決める。PI 制御(PID、PD 制御も含めて)を実装する時は、ゲインの調整を適切に決めることが重要になってくる。PI 制御の入出力信号の関係は以下の式であらわされる。

$$\Delta E = K_p\left\{e + \frac{1}{T_I}\int e dt\right\}$$

目標の回転速度を ω_r として、現在の回転速度 ω、目標値からの偏差($\omega_r - \omega$)を e とする．K_P, T_Iはそれぞれ比例ゲイン，積分時間である．

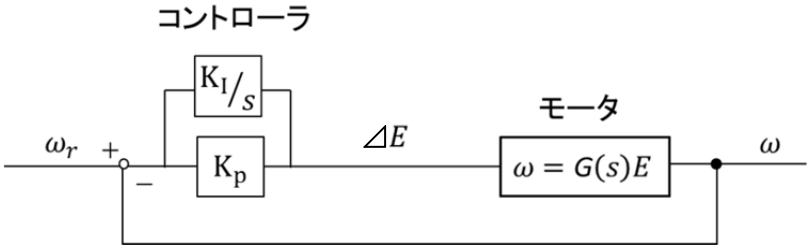


図11 DC モータの回転速度制御のブロック線図

ところで、本実験はデジタル制御システムであるため，連続時間システムを離散化する必要がある．そこで，⊿をサンプリング時間とし，上式の積分を矩形近似で近似するとそれぞれ次式の関係が成り立つ．

$$\int edt \fallingdotseq \sum_{i=1}^{k} e(k)\Delta$$

$$\Delta E(n) = K_P e(n) + K_I \sum_{i=1}^{n} e\,(k)$$

K_Iは積分ゲインである。$(K_I = K_P\,\Delta/T_I)$

今回の実験のプログラムでは、モータの印加電圧 denatu，目標回転速度 d_velocity，現在の速度 velocity，速度の偏差 err となっているので注意.

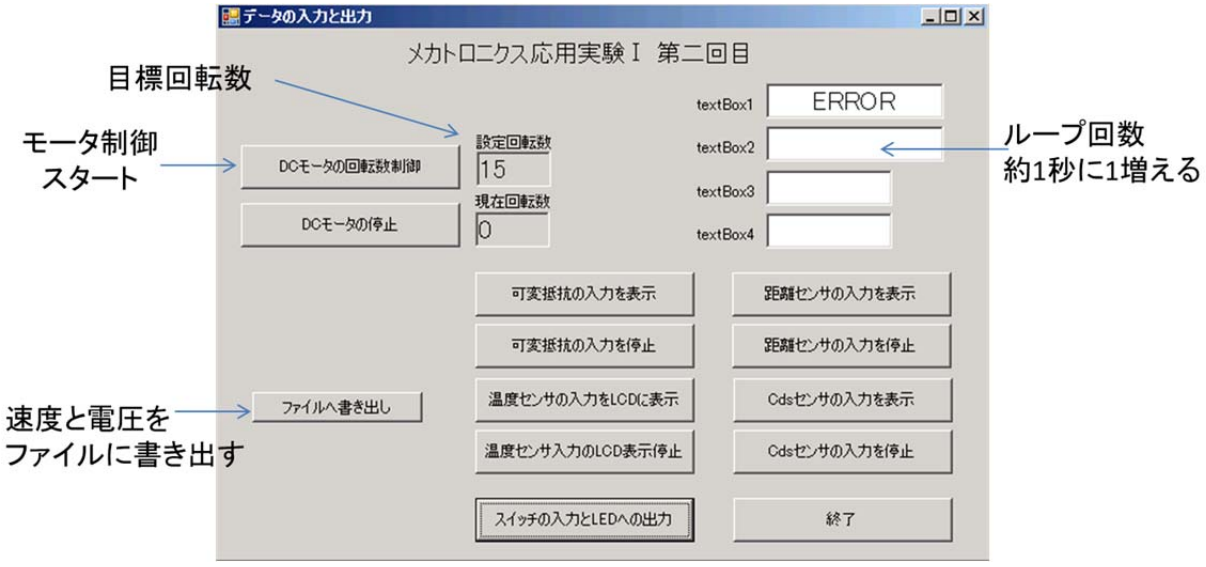
4.2.4 実験課題(2 週目)

1．プログラム内では，印加電圧用の変数 denatu (0≦denatu≦255)，目標回転速度 d_velocity=15[rps]，現在の回転速度 velocity，で定義されている．現在の回転速度を計測する部分は既に作られている．denatu に 40,60,80 のデジタル値を与えてプログラムを実行し，モータの回転速度等を観察しなさい.

2．C 言語と同様の if 文を用いて，現在の回転速度が目標速度より大きい場合は変数 d を小さく，逆の場合は変数 d を大きくするプログラムを作成し，回転盤の速度の挙動を観察しなさい.

3．P 制御を実現するプログラムを作成しなさい．比例ゲインを変えながらそれぞれについて実験しなさい。

4．PI 制御を実現するプログラムを作成し，グラフに出力しなさい．比例ゲイン、積分ゲインを変えながら、応答特性の違う複数のデータを取得しなさい.



5. レポートの構成

ⅰ．レポート表紙　（決められた様式以外は認めない）

ⅱ．実験の目的（テキストに書かれているが自分なりに書くこと）

ⅲ．原理（計測の原理，ＡＤ変換の原理，制御則など）

ⅳ．実験方法（分かりやすく簡潔に）

ⅴ．結果と考察
（第１週の各センサの計測結果のグラフ（４つ），第 2 週の各ゲインについてモータの速度(&電圧)のグラフをエクセルで描き、添付すること.

ⅵ．課題　*下記３課題は必須

a) 今回の距離センサは三角測量の原理を用いたものであったが､距離センサには他の原理を用いたものも複数存在する。これを調査し、その原理をまとめなさい.

b) 第 1 週目の可変抵抗と距離センサの計測結果について､量子化誤差はどうなっているか考察しなさい。必要なら、新たに量子化誤差のグラフを作成して説明してもよい。

c) 第 2 週目の(3)式について、計算過程を含めて導出しなさい。

d) メカトロニクス製品を一つ取り上げ,使用されているセンサ,アクチュエータを列挙し,その制御システムを調査し記述しなさい.

参考文献

深海登世司、藤巻忠雄　「制御工学(上)」　東京電機大学出版極
日本機械学会　「JSME テキストシリーズ　制御工学」　丸善出版
シータスク　「USB-EXE 取扱説明書」

6. 付録

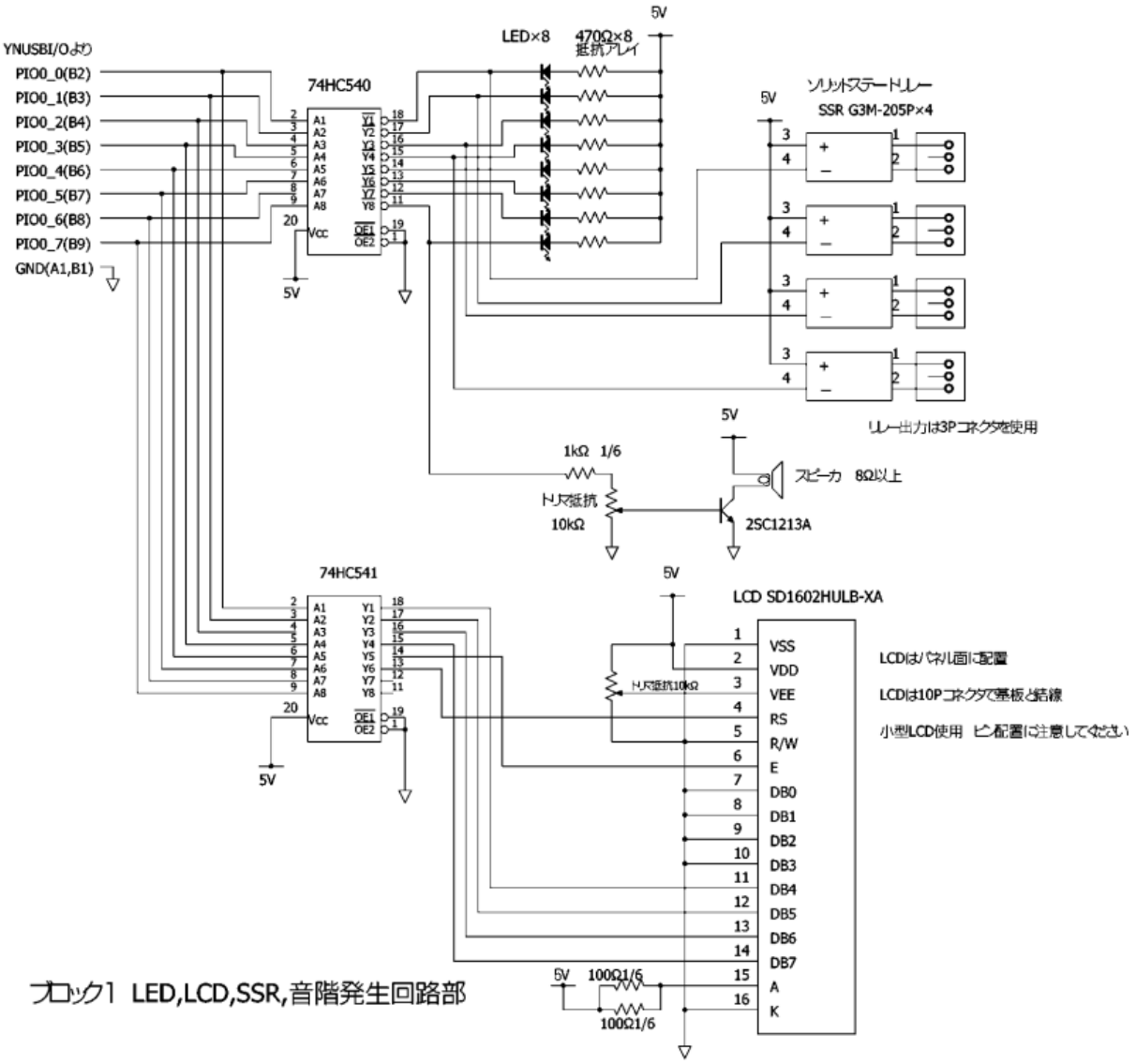
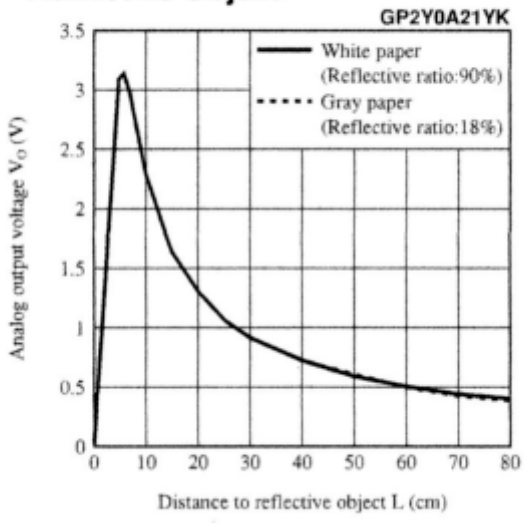
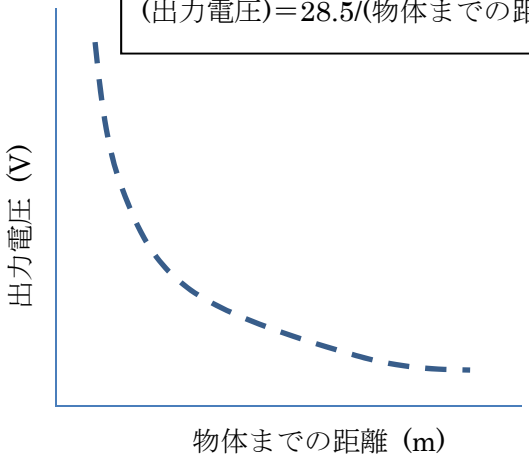
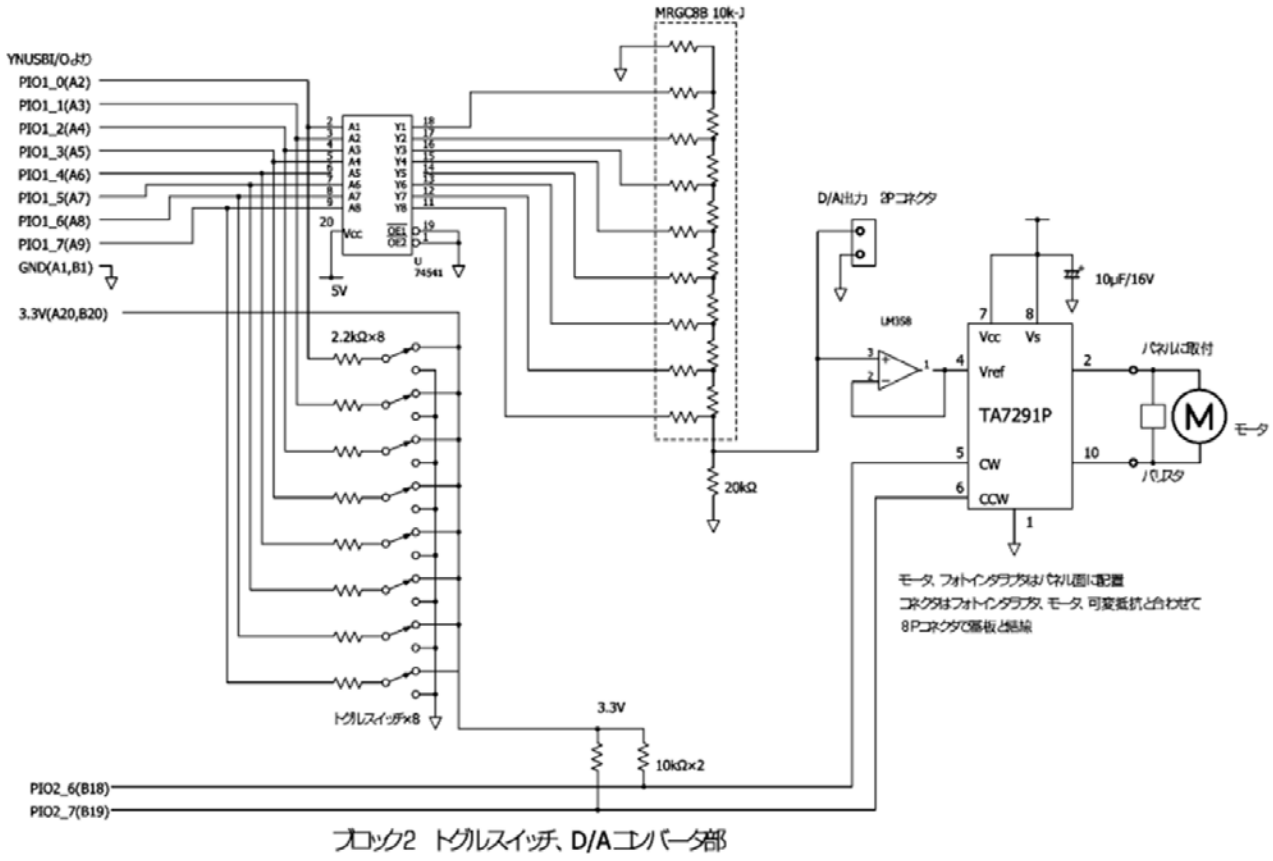


Fig.5 Analog Output Voltage vs. Distance to Reflective Object

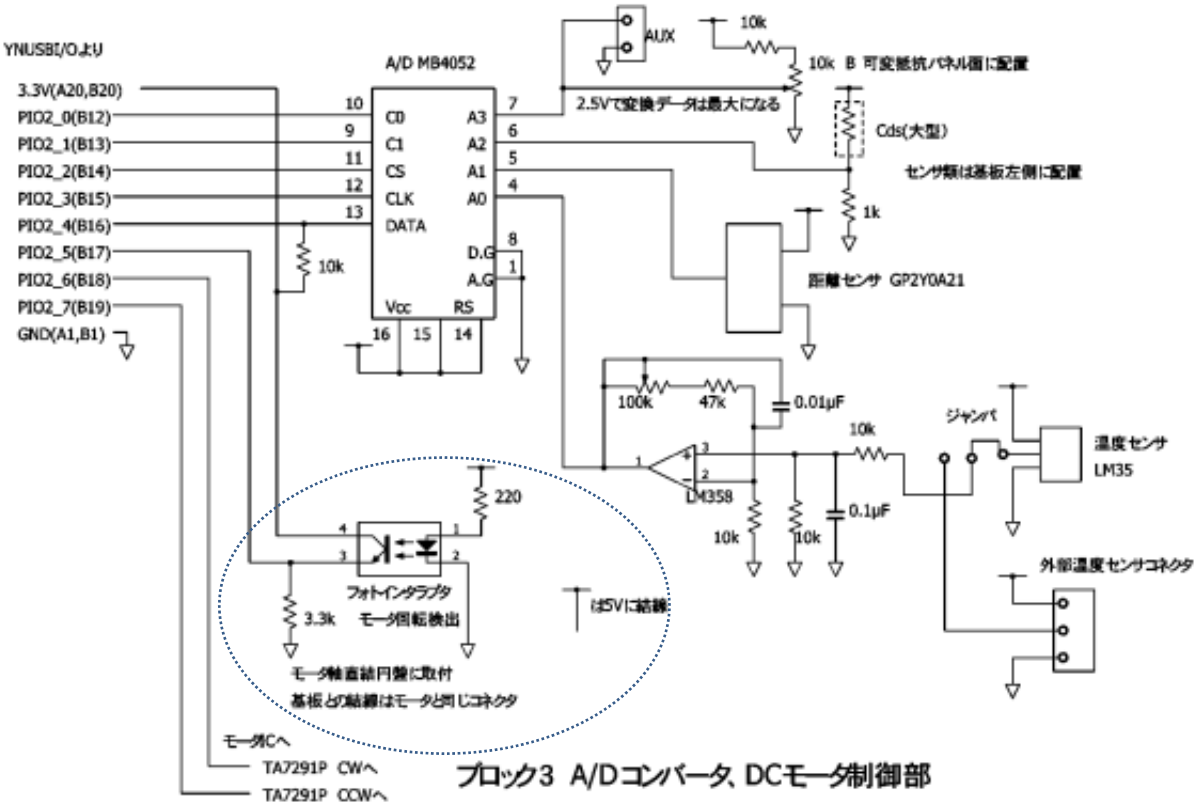


近似式
(出力電圧)＝28.5/(物体までの距離)



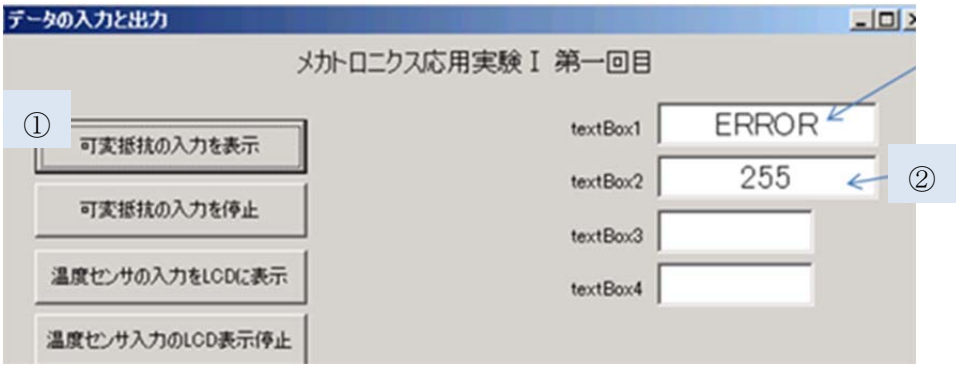


ブロック2 トグルスイッチ、D/Aコンバータ部



ブロック3 A/Dコンバータ、DCモータ制御部

１週目　AD 変換のプログラム（重要部分のみ抜粋）



①のボタンを押すと、次のプログラムが１回だけ実行される。

```
//////////////////////
//可変抵抗の入力を開始
//////////////////////
private void button5_Click(object sender, EventArgs e)
{
    timer2.Interval = 100;        //100ms 毎にタイマー2 が起動される。
    timer2.Enabled = true;        //100ms 毎にタイマー2 が起動される。
    ch = 0x03;                    //可変抵抗のチャンネルをセット
}
```

①のボタンを押すと、次のプログラムが 100ms 毎に１回、起動する。

```
////////////////////////////////////////////////////////////////
//タイマー２　（可変抵抗、距離センサ、または Cds センサの値を取得）
////////////////////////////////////////////////////////////////
private void timer2_Tick(object sender, EventArgs e)
{
    adcon(ch);              //関数 adcon を呼び出し v の値を更新する。
    textBox2.Text = Convert.ToString(v);      //ボックス②に v の値を表示。
}
```

関数 adcon は、AD 変換器から送られてきた 8 bit 信号を 10 進数に変換する関数である。
変換された後の 10 進数は、変数 v に格納されている。当然のことながら v の値は 0 から 255 の範囲内の整数である。　**＊実験課題においては、adcon の中は一切変更しなくてよい。**

```
////////////////////////////////////////////////////////////////
//　A/D 変換の関数（指定したチャネルの信号を A/D 変換するサブルーチン）
////////////////////////////////////////////////////////////////
public void adcon(int ch) //ch は変数、可変抵抗では 0x03、距離センサでは 0x01
{
    省略
}
```

課題１のヒント：タイマー２の中の、textBox2.Text = Convert.ToString(v);の(v)を
(v/7)のように 7 で割算し、プログラムを実行してみる。
すると、ボックス内に表示される値が、小さくなったはず。
ボリウムの抵抗値が 0 [kΩ]の時、出力電圧が 0 [V]、
10 [kΩ]の時、出力電圧が 2.5 [V]　になっているから・・・

２週目　DC モータ速度制御のプログラム（重要部分のみ抜粋）

```
////////////////////////////////////
//////////DC モータの制御

  private void button3_Click(object sender, EventArgs e)  //回転ボタンを押すとこの中の処理
  {
        pio1_ddr(0, 0xff); //DA コンバータ接続ポートを出力にイニシャライズ
        pio2_out(0, 0x80);//反時計回転開始 0x40(01000000)なら時計

              timer1.Interval = 500;              //500 ms 毎に Timer1 を起動する。
              timer1.Enabled = true;
                                                  //1 ms 毎にマルチメディアタイマを起動する。
              DgMmTimerCallback dgTimerCallback = new DgMmTimerCallback(TimerCallback);
      以下略
  }


//////////////////////////////////////////////////
/////////マルチメディアタイマー(回転速度を算出、Timercallback が 1ms 毎に呼び出される)
        private void TimerCallback(
        int timerID, int msg, IntPtr instance, IntPtr param1, IntPtr param2)
            {
              a = pio2_in(0) & 0x20;   //a はﾌｫﾄｲﾝﾀﾗﾌﾟﾀで穴の検知結果用の変数 0 なら板、1 なら穴がきている
                 if (a == 0)          //もし板なら
                 status = 1;         //status を 1 とする

                 if (a != 0)          //もし穴なら
                 {
                       if (status == 1)   //さらに直前が板であったなら
                       {
                            velocity++;     //velocity に 1 を加える
                            status = 0;   //staus2 を 0 に戻して 2 重に数えないようにする
                       }
                 }
            }


//////////////////////////////////////////////////
//////////タイマー１（加減速を行うタイマイベント）500ms 毎に呼び出される

            private void timer1_Tick(object sender, EventArgs e)
            {
                        counter++;//100ms 毎に 1 増える

                        //PI 制御による電圧 d を決定するプログラムを書く
                        //d_velocity は目標速度、velocity は現在の速度,
                        //四捨五入は　0.5 加算して Math.Floor
      ////   ここにプログラムを書く。  ///
        ┌──────────────────────────────────────┐
        │ ココに追加する              ↑↓のプログラムはそのままにしておくこと │
        └──────────────────────────────────────┘
            if (denatu >= 255) denatu = 255;//d が 255 を超えたら強制的に 255 にする
             if (denatu <= 0) denatu = 0;
             pio1_out(0, denatu);//モータに電圧 denatu をかける

            v_buff[counter] = velocity;//ファイル出力のため配列に格納
            d_buff[counter] = denatu;//ファイル出力のため配列に格納
            textBox6.Text = Convert.ToString(velocity);//テキストボックスに表示
            textBox2.Text = Convert.ToString(counter);
            textBox6.Update();
            textBox2.Update();
            velocity = 0;          //velocity を 0 に戻す
}
```